Path - Planning for Unmanned Air Vehicles

Scott A. Bortoff
Visiting Scientist, AFRL / VAAD, Summer 1999

Associate Professor
Department of Electrical and Computer Engineering
University of Toronto
Toronto, Ontario Canada
Phone: (416)-978-0562
Email: bortoff@control.toronto.edu

August 15, 1999

Abstract

Unmanned air vehicles of the future will be more lethal and more autonomous than the remotely-piloted reconniassance platforms in use today. Among the many open issues in their development is that of path planning: Without a pilot, computer algorithms must be developed that generate a flight path in real time. This is a challenging problem for several reasons. The algorithm must compute a stealthy path, steering the aircraft's radar signature around known enemy radar locations. These paths must be both minimal-length and feasible for the aircraft to follow. The algorithm must allow for coordination among multiple UAVs. And it must run in "real-time," since enemy threats can change during a mission, forcing a path re-plan. Finally, it must be memory and computationally efficient, since it will run on an airborne processor.

In this report, we investigate three different approaches that can be used to generate desirable UAV paths. These include 1) Graphs, which represent the path as a sequence of edges of a graph, 2) Optimal Control, which computes an optimal path for a given cost function, and 3) Virtual Potential Fields, which compute a path as the solution to a related, simpler problem. Our treatment of graphs is summary; the ideas are introduced, and simple graph structures are used as initial conditions for the latter two approaches. For the optimal control approach, we define a conservative set of UAV kinematics that the generated path will satisfy, and then construct a cost function that captures the important characteristics of the problem. We solve the optimal control problem for some simple cases, and set-up the more general solution as the solution to a two-point boundary value problem. We then turn to the virtual potential field approach, describing it in detail and providing some simulation examples. We close this report with a comparison between the approaches, and some recommendations on the future of UAV path planning research.

Acknowledgements

I would like to thank Siva Banda, Phil Chandler, and the rest of the "Summer 1999 UAV Team:" Rajeeva Kumar, Tim McLain, Meir Pachter, Steve Rasmussen, Cory Schumacher, and Sahjendra Singh. The twelve weeks I spent at WPAFB were very fruitful, and I enjoyed the company of everyone in the group. It went by all too quickly. I would also like to thank the US Air Force and Ball Aerospace for their financial support under the Visiting Scientist Program. Finally, I thank the University of Toronto for granting me this sabbatical leave.

Scott A. Bortoff
August 15, 1999
Dayton OH

List of Symbols

Term	Description
E_0	Inertial reference frame, attached to the Earth.
(x_0, y_0, z_0)	Inertial reference frame cartesian coordinate axes.
E_b	UAV "body" reference frame. attached to the UAV center of mass.
(x_b, y_b, z_b)	Body frame cartesian coordinate axes
h	Height of UAV, assumed to be constant
$(x, y, 0)$	Cartesian coordinates of UAV, expressed in E_0 frame
ϕ	Roll angle of UAV
ψ	Heading angle of UAV
$(\bar{x}_k, \bar{y}_k, h)$	Cartesian coordinates of k^{th} radar site, expressed in E_0 frame
$(\hat{x}_k, \hat{y}_k, \hat{z}_k)$	Cartesian coordinates of k^{th} radar site, expressed in E_b frame
$(\hat{\mu}_k, \hat{\nu}_k)$	Spherical coordinates (elevation, aximuth) of k^{th} radar site, expressed in E_b frame
$\sigma(\hat{\mu}_k, \hat{\nu}_k)$	Radar energy intensity, expressed in spherical coordinates relative to frame E_b.
$s(x, y, \psi, u, \bar{x}_k, \bar{y}_k)$	Radar energy intensity, expressed in cartesian coordinates in the E_0 frame.
$r(x, y, \bar{x}_k, \bar{y}_k)$	Distance between UAV and k^{th} radar site, raised to the fourth power.
$J(u)$	Cost function.
N	Number of radar sites.
Q	Radar returned energy weight.
R_1, R_2	Turning (control) weights.
M	Number of masses in the potential fields approach.
b	Damping term in the potential fields approach.
κ	Spring constant in the potential fields approach.
m_k	Mass used in the potential fields approach.

1 Introduction

For several decades, the US Air Force has used unmanned air vehicles (UAVs) for little more than remotely-piloted reconniassance platforms and target practice drones. But their role is about change. UAVs of the future will be able to accomplish a mission, such as delivering ordinance to a target, autonomously — with little or no human intervention. They will be able to sense a change in their environment, such as a threatening missile, and alter their mission plan. Multiple UAVs will be able to coordinate the timing of an attack. They will be capable of independently seeking and destroying targets of opportunity. And of course, they will continue to serve as reconniassance platforms.

Among the many open issues in their development is that of *path planning.* A path planning algorithm computes a trajectory from the UAV's present location to a desired future location, e.g. a target. But the path planner must do more than simply generate a set of way points along the straight line connecting the present location with the target. A good path planning algorithm must possess several important attributes, making its design a multiple-objective optimization problem. First, and most importantly, it must compute a *stealthy* path, steering the aircraft and its radar signature around known enemy radar locations. No aircraft scatters or reflects radar radiation uniformly in all directions. Rather, radiation is radiated more strongly in some directions, and less strongly in others. The path planning algorithm should take advantage of "notches" in the radar signature, pointing them toward known enemy radar sites. Conversely, directions which radiate strongly should be turned away from these sites. Second, generated trajectories should be of minimal length, subject to the stealthy constraint. Of course, UAV range is limited, and time spent over enemy territory should be minimized, so path length should always be a factor in any algorithm. Third, the trajectory should be feasible for the aircraft to follow. Any trajectory that causes the aircraft to fly at velocities outside its flight envelope, or to turn at an impossible rate, is obviously not acceptable. Fourth, the path-planning algorithm must be compatible with the cooperative nature envisioned for the UAV. A typical mission might involve multiple UAVs attacking single, well-defended target. The path-planner would be a component — a subroutine — of the hybrid control system that ensures all UAVs arrive simultaneously. And finally, path-planning algorithms are expected to be coded in software that runs on an airborne processor. Thus, they must be computationally efficient and "real time," enabling the UAV to re-plan its trajectory should an unforeseen threat arise. This last characteristic should not be dismissed as an implementation issue. Existing path planning methods that are extremely computationally complex should be ruled-out as potential solutions early in the design cycle.

Fortunately, the problem of trajectory planning has been studied for decades in a variety of different contexts. The aerospace path planning problems of the 1960's were solved largely through the application of the calculus of variations, itself invented hundreds of years earlier, e.g. [1, 2, 3]. The development of industrial robotic manipulators in the 1970s and 1980s encouraged researchers to study new path planning methods, largely motivated by collision avoidance. More recently, the focus in robotics has shifted to path planning for autonomous mobile robots. Although most of this work concerns wheeled vehicles, e.g. [4], path planning algorithms have been developed for UAVs as well as underwater robots.

In this report, we explore three different solutions to the path planning problem for

UAVs. First, we briefly outline the so-called graph-based methods. These approaches "grid" the airspace, representing the set of all possible paths to a target as the edges of a graph. The graph can be searched for an optimal path using well-known methods such as Dijkstra's Algorithm to compute an optimal trajectory. Next, we look at an optimal control approach. We propose a simple kinematics model of an aircraft, and derive a meaningful cost function that captures the essence of the path planning problem. This is a weighted sum of path length, returned radar energy, and expended control energy. The numerical solution of this problem is also investigated. Finally, we solve the the path-planning problem using a method that makes use of classical mechanics and virtual potential fields. This represents the path as a chain of masses connected by springs and dampers. The masses are acted upon not only by the restoring spring force, but also a virtual repulsive force pushing from each radar site. The physics of the situation force the chain to bend around the radar sites, minimizing a weighted sum of average distance from the radar sites and path-length. The idea is to simulate the mechanical system on-line, and use the final configuration of the masses as way-points.

This report is organized as follows. In Section 2, we look at the key criteria that a successful algorithm must possess: Stealth, Dynamic Constraints, Cooperation, and Efficiency. In Section 3, we briefly examine graphical methods of solution. Our presentation of this approach is by no means complete. Rather, our aim is to show the strengths and weaknesses of a graphical approach when applied this particular problem. In Section 4, we turn to an optimal control approach. We propose a simple dynamic model of a UAV, and then derive a meaningful cost that captures the critical design issues mentioned above. We then partially solve the problem using well-founded methods in optimal control theory and the calculus of variations, resulting in a set of two-point boundary value problems that require numerical solution. In Section 5, we explore the virtual potential field method, providing some simulation results for some benchmark problems. Finally, in Section 6, we draw some conclusions, comparing the different approaches, showing how they can be combined into a hybrid path planner. Of course, we also make recommendations for future research.

2 Path-Planning Algorithm Design Issues

A successful path-planning algorithm must produce trajectories that are stealthy, have minimal path length, satisfy the UAV dynamic constraints, and are compatible with the cooperative missions envisioned for UAVs. And it must be efficient enough to run on an airborne processor. These design criteria are detailed in this section.

2.1 Stealth

It is well-known that today's military aircraft are designed to minimize the probability of detection by enemy radar. They do this by absorbing incoming radar radiation and/or reflecting it in a direction different than the ambient direction, so that little is reflected back to the original radar site. Although great efforts are made, the fact remains that all aircraft will reflect some small amount of radar radiation back to a radar site. The only way to avoid detection is to ensure that the UAV trajectory is designed such that worst-case reflected radiation is below a detection threshold.

If an aircraft reflected radar energy uniformly in all directions, the path-planning would be relatively trivial. (This will be made clear later in this report.) Unfortunately this is not the case. Rather, because aircraft are designed to redirect incoming radar radiation in certain safe directions, while minimizing reflected radiation in others, its radar "signature" is *by design* not uniform in all directions. For example, it is well-known that the F-117 "stealth fighter" is shaped such that incoming radiation is not reflected back in its incident direction, but rather is reflected in different directions, causing little to be returned to the enemy radar site. This means that, in general, the radar signature for a given aircraft will have "notches" and "spikes" in known directions. It is important to generate trajectories that steer the notches toward enemy radar sites, while steering the spikes away from them. This significantly complicates the path planning problem.

In this study, we will assume that the UAV does not *reflect* radar energy, but rather *radiates* energy on its own. This simplifies the problem because we don't have to consider changing angles of incidence and reflection along a flight path. We will assume that the amount of radiated energy depends on direction in a pattern that is easiest to express using a spherical coordinate system. To do so, we first attach the so-called "body frame" to the UAV according to convention, such that the x_b-direction points in the direction of flight, the y_b-direction is along the right wing, and the z_b-direction points "down." This is shown in Figure 1. Let μ denote the azimuth angle, measured with respect to the x_b-direction, and let ν denote the "elevation" angle, measured with respect to the z_b axis, as shown in Figure 1. Note that ν is not measured with respect to the $x_b - y_b$ plane, in order to simplify the formulas that follow. With this convention, we can define a function $\sigma(\mu, \nu)$ which represents the amount of radiated energy as a function of direction. For example, the function

$$\sigma(\mu,\nu) = \begin{cases} 4\cos^2(3\nu) & \nu \leq \frac{\pi}{6} \text{ rad} = 30^\circ \\ 4\cos^2(3\nu)\sin^2(2\mu) & \nu > \frac{\pi}{6} \text{ rad} = 30^\circ \end{cases} \tag{1}$$

which is plotted in Figure 2, has a highly-directionally dependent "signature" consisting of five lobes. In these directions, the UAV would be highly visible, while in other directions, the radiated energy tends to zero. This particular function is chosen because of its particular

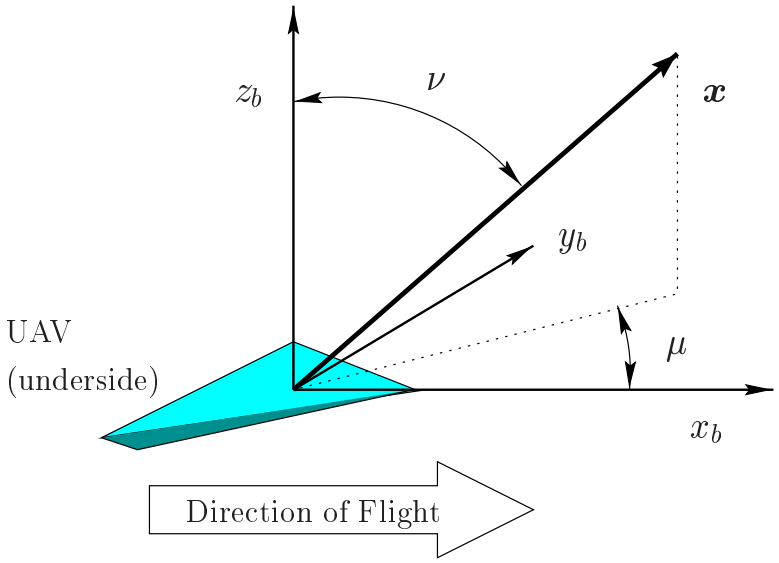

Figure 1: UAV body frame, showing the x_b, y_b and z_b axes, and the definition of the spherical coordinates ν and μ for an example vector $\boldsymbol{x}$.

computational simplicity, not necessarily because it represents the radiation pattern of any particular aircraft. Nevertheless, it does capture the nature of the problem: To steer the UAV such that the radiation lobes stay away from known enemy radar locations. We will use this function throughout this report.

2.2 Minimal Path Length

In every path-planning algorithm, be it a graph, an optimal control problem, or a solution obtained using virtual forces, a cost is minimized. The cost is a weighted sum of terms. One of these terms will represent energy received by each radar site. Another will represent path length. This is important to include for reasons that are more or less obvious. Certainly minimizing path length is important in order to conserve fuel. It is also important to reduce the amount of time spent over enemy territory. What may not be obvious is that path length must be included in the cost because otherwise the optimization problem will not be well posed. That is, if a cost contains only a penalty on radar detection, the "optimal" path would be to stay at home, giving a cost of zero. If constraints are used to force solutions that include a target, then the "optimal" path might fly the UAV far away from any enemy radar sites, giving an excessively long (or, in the limit, infinitely long) path. So path length must be included in the cost to make it a well-posed optimization problem. This is not unlike the usual requirement that the weight on the control be positive ($R > 0$) in a classical (LQR) optimal control problem.

2.3 Dynamic Constraints

The path generated should be "flyable" by the UAV. A path that requires the UAV to reduce its airspeed below a stall condition, or to turn at an excessive rate, will not be acceptable.

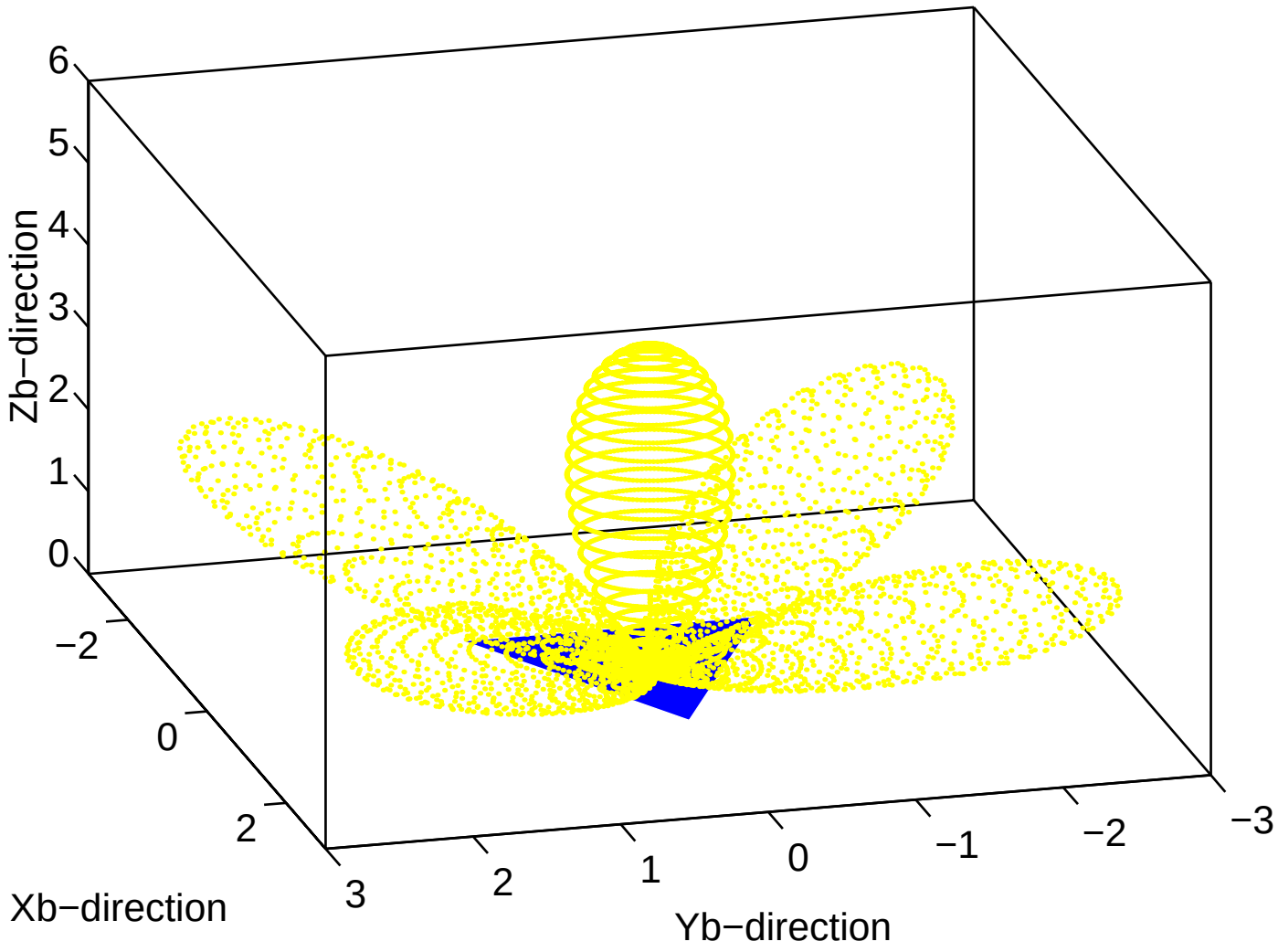

Figure 2: An example UAV radar signature used in this study, defined in equation (1).

Thus, a simple model incorporating the dynamic constraints will somehow have to be imposed on the optimization problem. In an optimal control setting, this is relatively straightforward: The dynamic constraints can be included through use of Lagrange multipliers (costates). In a graphical approach, we simply assign the vertices of the graph at points in space through which it is possible for the aircraft to fly, and then connect these vertices with edges, along which the aircraft can fly. Thus, in this approach, any path through the graph is "flyable."

That said, the path-planning problem is not an "inner-loop" control problem. The model of aircraft dynamics to be included must be the simplest possible, capturing only the essential constraints of flight. If an overly detailed model is incorporated, the problem will increase in computational complexity, becoming impossible to solve in real-time. This point can be understood by analogy. Suppose we are planning an automobile trip from Dayton OH to Hartford CT. The path planning problem in this context is to determine the sequence of highways to take. We want to minimize a weighted sum of a measure of our travel time and our probability of receiving a traffic ticket for speeding. (Assume we know where all of the

police are located.) To solve this problem, we require a map, and we need to know the average speed of the car. Then dynamic programming can be used to solve this problem efficiently. Importantly, this solution does not require knowledge of the detailed of the dynamics of the car, only the average velocity. That is because we are not concerned with the details of the trajectory, such as lane changes along the way. The UAV path planning problem is at the same level. We are not interested in control surface settings, only in generating a flight path, or set of way points, along which it is possible for the UAV (and its lower-level controller) to track. Thus, we shall propose a very simple *kinematic* model of UAV "dynamics," which includes limits on roll and turn rates.

2.4 Cooperation

Using UAVs in a cooperative sense to accomplish a mission is one vision of their use. For example, a group of three UAVs might be assigned to attack a target simultaneously from different directions. In such a scenario, each UAV would plan a trajectory that has the same final state and time. Should something such as a missile threaten one of the UAVs, each of these paths might require modification. So, the team's final time, or ETA, is not necessarily defined a priori. It must be *defined* as part of the coordination effort. ETA determination will use the path-planner as a subroutine. A good path planning algorithm must be consistent with this vision. Specifically, it should provide a low-observable path (one minimizing a well-defined cost function) for a *given, fixed* final time, and it should also provide a path and a final time when the latter is not *a priori* specified. We should think of the path planner as a subroutine, to be executed repeatedly by a higher-level mission controller, in order to determine UAV and team ETAs in addition to the path itself. All three methods examined in this report are well-suited for both the free and fixed final time problems.

2.5 Real-Time Performance

The path planning algorithm must be both memory and computationally efficient, since it will be realized in software that runs on an airborne processor. It must also be capable of re-planning a path should a new threat arise. Thus, it would be an advantage if a planning algorithm could make use of "old" information in addition to the new threat information when doing re-planning. For example, adding a vertex to a graph does not necessarily mean the entire graph needs to be re-searched from scratch. The data structure used to represent the graph should be amenable to adding and deleting vertices. Finally, the algorithm must scale well. This means that its computational complexity scales as a polynomial and not an exponential in the number of radar sites or UAVs, for example. What is appropriate for ground-based, pre-mission path planning, which would use essentially unlimited processing power, may not be appropriate for an airborne deployment.

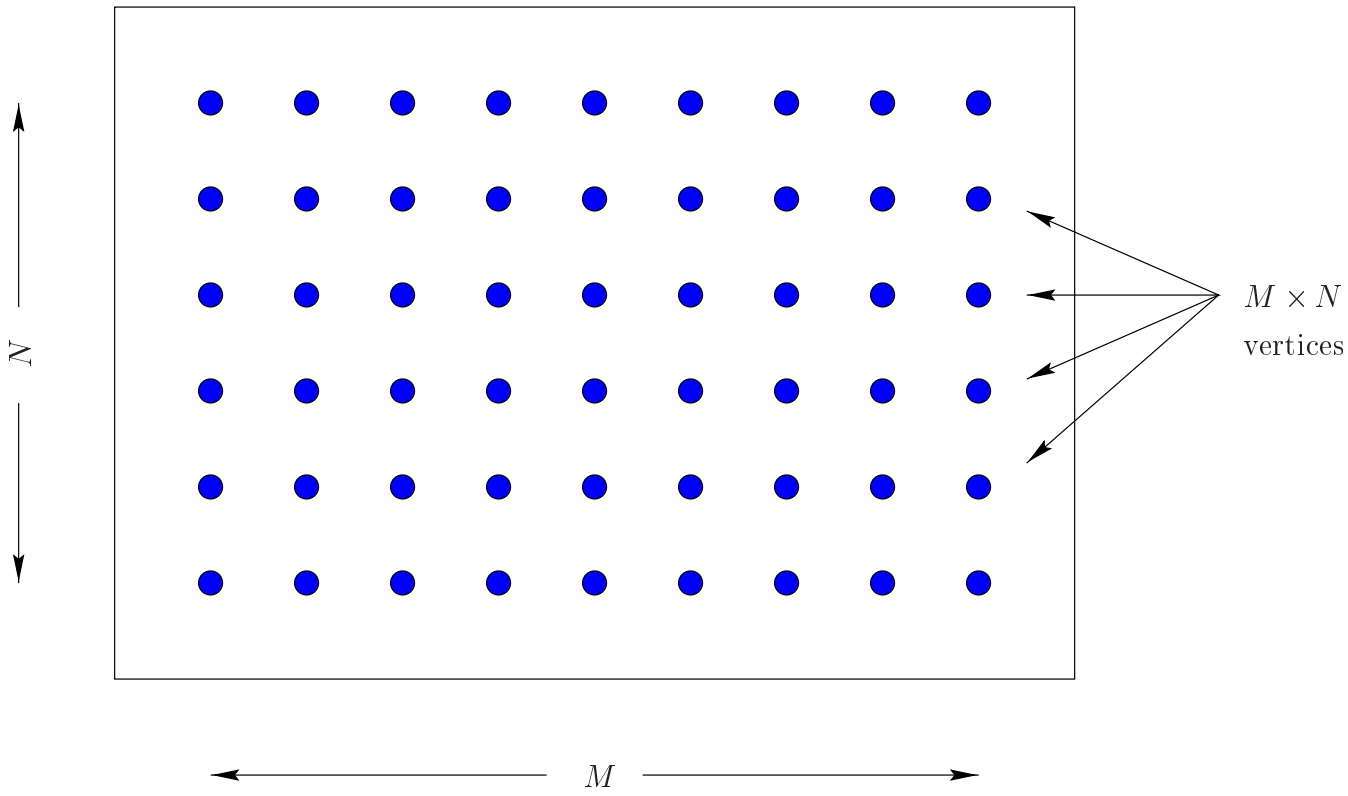

Figure 3: Vertices of a graph.

3 Graph Approaches

A graph is a set that contains vertices and edges. Graphs can be used to solve the path planning problem, just as they do for many similar robotic path-planning problems, by assigning vertices to discrete points of state space, connecting them appropriately with edges, assigning weights (costs) to each edge, and then searching the graph for an optimal trajectory using one of several well-known algorithms.

The basic idea can be illustrated by example. Suppose we have a rectangular area of space through which we wish to plan a trajectory. We first quantize the configuration space of the UAV. Assume that it can be located only at regularly spaced points on a horizontal Cartesian grid, as shown in Figure 3. (For simplicity, assume that it flies at a constant height h, although the z-direction could also be quantized.) To be more specific, let us assume there are $M \times N$ discrete locations in the x and y directions, respectively. Now, at each of the $M \times N$ locations in the plane, suppose that the UAV can assume a number P of discrete orientations (headings), spread uniformly about the compass directions. In total then, the UAV can assume $M \times N \times P$ discrete configurations, each of which is a vertex on the graph.

The next step is to connect the vertices using edges. Here, we account for the dynamics of the UAV, in a somewhat course manner. We will assume that when the aircraft is at a particular vertex, then it can proceed along the same direction, turn left, or turn right. Thus, we will connect each vertex with six other vertices (3 "input" and 3 "output"). This is diagrammed in in Figure 4. For this example, we have a total of $24 \times M \times N$ edges.

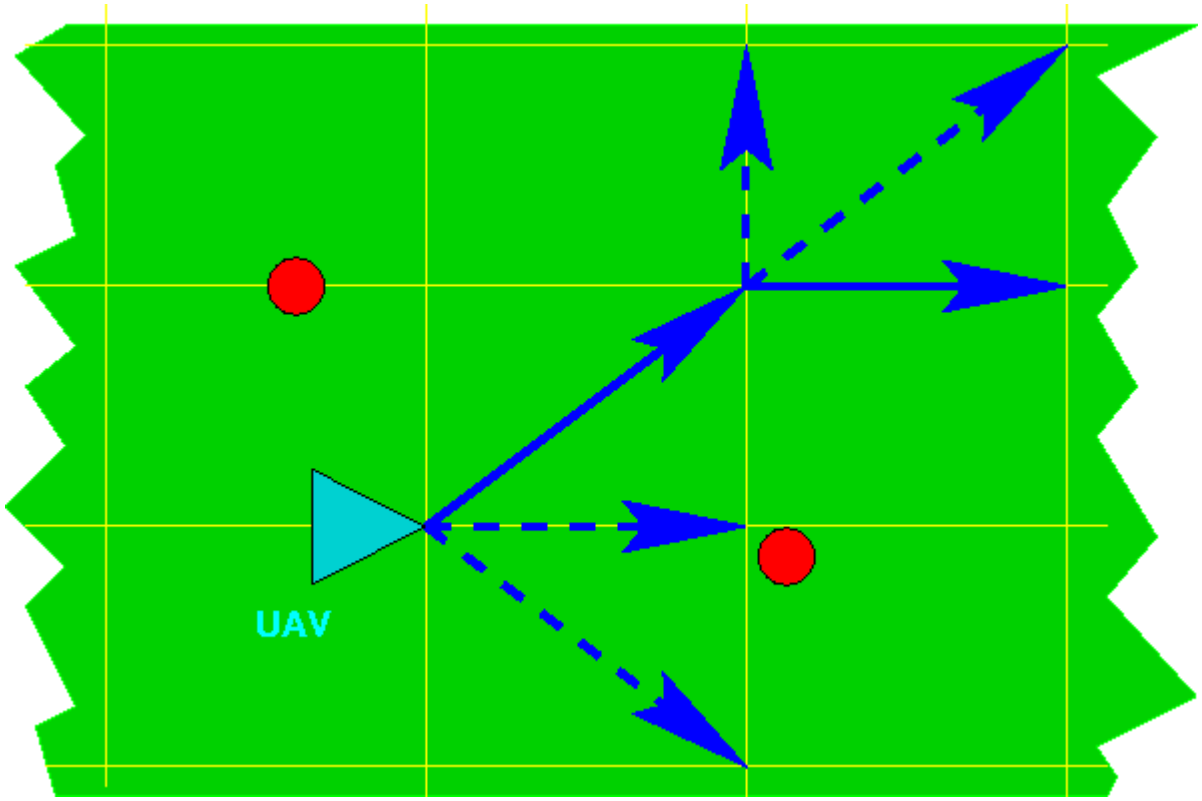

Figure 4: Edge assignment of a graph.

The next step is to assign a cost or weight to each edge. This is a nonnegative number that reflects the cost for the UAV to move along that edge, and is a weighted sum of the edge path length and probability of radar detection along the edge. (Costs are discussed at length in Section 4 in the context of optimal control. The costs used there are also applicable to the graph approach.) Once costs are assigned for each edge, a method such as Dijkstra's algorithm [5] can be used to search the graph for the minimum cost path between any two vertices. If V is the number of vertices and E is the number of edges, the computational complexity of Dijkstra's algorithm is $O(V \log(V) + E)$. Since in our example, $E = 3V$, meaning the graph is sparse, this estimate is dominated by the $V \log V$ term.

A graph approach offers a number of advantages over other methods such as optimal control. First, it gives a global solution, although it should be remembered that it is global only on the set of discrete positions (vertices) and edges. Second, the amount of calculation required to compute the optimal is bounded. This bound may be large, but the fact that it exists at all is extremely useful for any real-time implementation. (Recall that "real time" doesn't necessarily mean "fast," but rather "predictable.") The major disadvantage is the fact that the solution may be overly quantized. If the radar signature has a large number of nodes, then the graph will have to have a large number of orientations (more than just the 8 used above), and this will significantly increase the number of vertices and edges. Moreover, a full three-dimensional problem will increase the number of vertices and edges exponentially. For a realistic problem, this increase might make the whole approach infeasible.

Of course, there are heuristic algorithms that can be used to search graphs not in an

exhaustive way but by pruning away parts of the graph that will probably not yield an optimal solution. An excellent example of using graphs to generate trajectories using the so-called A^* algorithm can be found in [4]. But these methods are beyond our scope.

3.1 Delaunay Triangulation

Another way to curtail the exponential increase in computational complexity of the graph method is to use a sequence of graphs. We start with a relatively course graph, one with a small number of vertices, and search it for an optimal solution. We then build a new graph in a neighborhood of this optimal solution, and search it for a new optimal solution. The second graph would have a higher density of vertices and edges. This procedure can be repeated. In this way, we have a more detailed level of quantization for the final trajectory, while reducing the amount of computation required by the graph search procedure.

Taking this approach to its extreme, we might begin by constructing the simplest possible graph that captures the very essence of the problem. Such a graph can be constructed using Delaunay triangulation and its geometric dual, Voronoi polygons. This procedure, which is used in many different fields, including computational fluid dynamics, computer graphics, and statistics, begins with complete knowledge as to the number and location of each radar site, as illustrated in Figure 5 (top). For every triplet of radar sites, there exists a unique circle that passes through all three. Consider only those triplets whose circle does not enclose any other radar sites, as shown in Figure 5 (bottom). The set of all such triplets is called the Delaunay triangulation, and the centers of the circles are called Voronoi points. We may now construct a graph by defining the vertices as the Voronoi points. Edges are drawn to connect two Voronoi points if and only if their associated Delaunay triangles share an edge. By drawing all such edges, we construct the Voronoi diagram or graph. The edges of the Voronoi diagram have the property that they are equidistant from pairs of radar sites.

If we consider the Voronoi diagram to be a graph, with vertices being the Voronoi points and edges being the connecting segments, then we can assign weights to each edge just as we would for any graph. Searching this graph is done exactly the same way any graph is searched. This will produce an optimal path from the set of Voronoi segments which is the simplest path through the radar sites, in the sense that it "tells" the UAV which pairs of radar sites to fly between. While this approach might provide an overly simplified, coarse flight path, it is useful as an initial condition for other methods that can capture more of the detail inherent in the path planning problem, such as a more refined graph or the optimal control approach outlined in the next section.

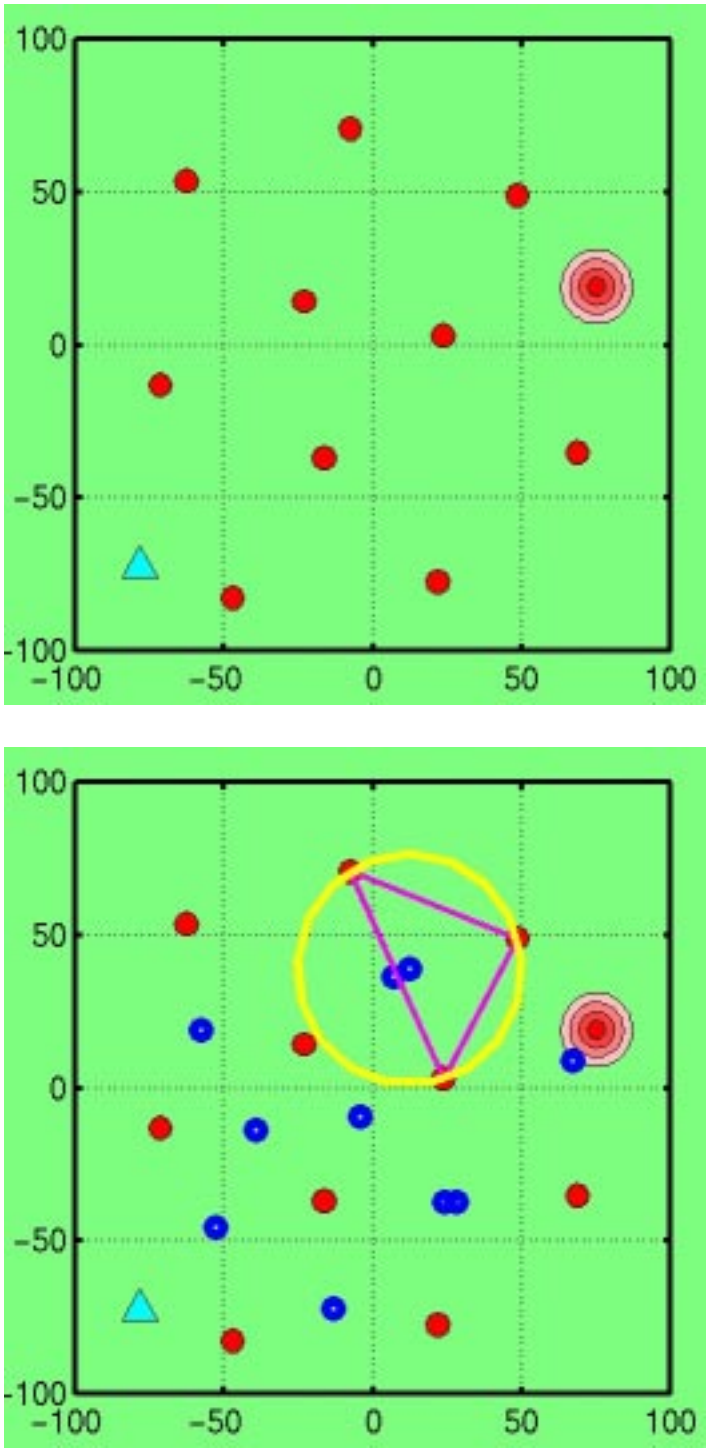

Figure 5: Top: Delaunay triangulation begins with knowledge of the ten radar sites (small circles), the UAV (triangle) and target (concentric circles). Bottom: Triplets of radar sites are selected such that the unique circle that passes through them contains no other radar sites. The center of each circle is a Voronoi point.

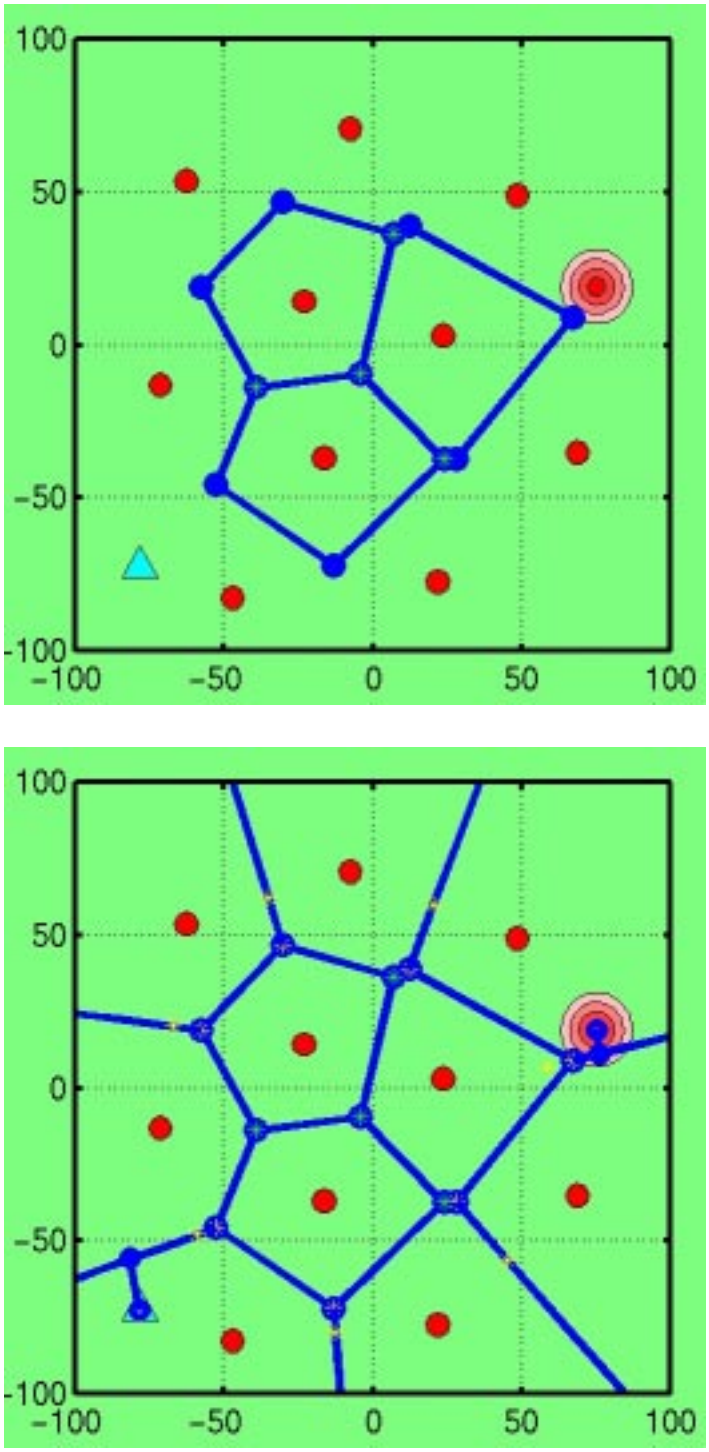

Figure 6: Top: Delaunay triangulation is continued by connecting the Voronoi points. Two points are connected if their associated Delaunay triangles share an edge. Bottom: The Voronoi diagram is completed by extending rays from the edge Voronoi points midway through the appropriate Delaunay triangle edge. The target and UAV are attached, weights are assigned, and the graph is searched.

4 Optimal Control

In this section we frame the path planning problem as a classical optimal control problem of the form

$$\min_u J(u) = h(t_f, \boldsymbol{x}(t_f)) + \int_{t_0}^{t_f} l(\boldsymbol{x}, u)\, dt \tag{2}$$

subject to the constraints

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, u), \quad \boldsymbol{x}(t_0) = \boldsymbol{x}_0, \quad \boldsymbol{x}(t_f) = \boldsymbol{x}_f. \tag{3}$$

Here, $\boldsymbol{x}(t)$ represents the state of the aircraft at time $t \in [t_0, t_f]$, $u(t)$ represents a control input (to be defined below), $\boldsymbol{x}_0$ and $\boldsymbol{x}_f$ are the specified initial and final states, respectively, t_0 and t_f are the initial and final times, respectively, $h(t_f, \boldsymbol{x}_f) \geq 0$ is a penalty on the final state, and the integrand $l(\boldsymbol{x}, u) \geq 0$ penalizes the trajectory and control.

Our goal is to define meaningful expressions for $h(t_f, \boldsymbol{x}(t_f))$, $l(\boldsymbol{x}, u)$ and $\boldsymbol{f}(\boldsymbol{x}, u)$ in (2) and (3), and then to solve the resulting optimal control problem for $u(t)$ for both the fixed final time problem (when t_f is specified) and for the free final time problem (when t_f is unknown). We begin by defining $\boldsymbol{f}$ to be a simple model of the aircraft kinematics.

4.1 Dynamic Constraints

Before deriving a simple model of UAV dynamics, we make the following assumptions.

1. The UAV flies at a fixed altitude h. This is a reasonable assumption, made primarily to simplify the numerical aspects of the problem. It would be a straightforward exercise to relax this assumption, allowing the UAV to change altitude according to a second control input.

2. The UAV flies at a fixed, constant speed, which without loss of generality is normalized to 1. For purposes of path planning, this significantly simplifies the problem. This is because the path length must be part of the cost (2). If a path is parameterized by t and is expressed in Cartesian coordinates $(x(t), y(t))$ then its length is defined as

$$\text{path length} = \int_{t_0}^{t_f} \sqrt{\dot{x}^2(t) + \dot{y}^2(t)}\, dt. \tag{4}$$

   So the integrand of (4), with $\dot{x}$ and $\dot{y}$ replaced with the right-hand-side of (3), would be part of $l(\boldsymbol{x}, u)$ in (2). However, if the dynamics have a constant velocity normalized to one, then (4) is equal to $t_f - t_0$ and $l(\boldsymbol{x}, u)$ is significantly simplified.

   Note that this assumption is made only for the purposes of *planning* the trajectory. Once the path is planned, the UAV may change its speed when flying along the trajectory, in order to satisfy other constraints such as meeting a team ETA.

3. All of the radar sites are at sea-level, simplifying some of the expressions that follow. Incorporating surface features would not complicate the problem at all, as long as the radar altitudes were known.

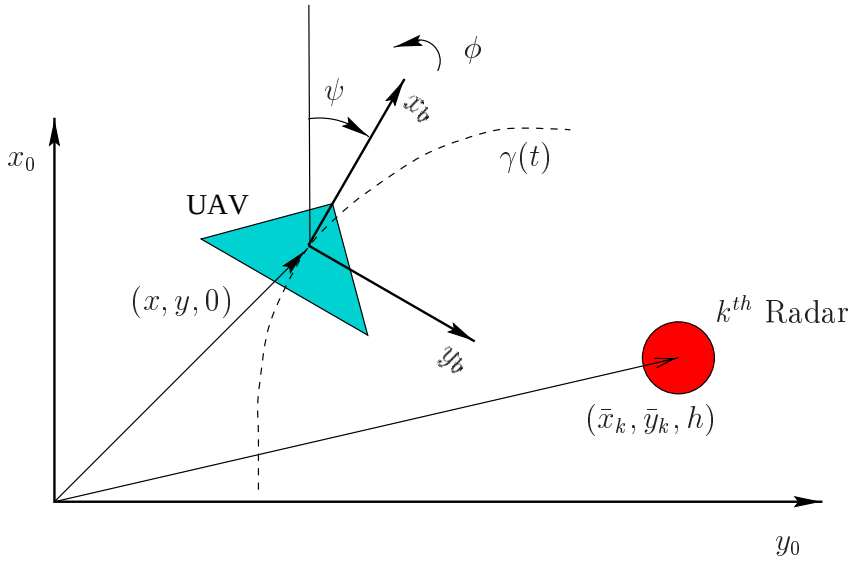

Figure 7: Frames of reference.

Before proceeding, we introduce two important frames of reference. Referring to Figure 7, let E_0 denote an earth-fixed inertial frame with axes labeled x_0, y_0 and z_0, respectively. This frame is oriented such that its origin is a fixed altitude h, and the z_0-axis points down toward earth's surface. Next, we denote the frame of reference that is attached to the UAV (the so-called body frame) as E_b, and label its axes x_b, y_b and z_b. Its origin is located at the UAV center of mass, and it is oriented such that x_b is aligned with the UAV velocity vector, y_b points out the right-hand side of the aircraft, and z_b points down toward the ground, as illustrated in Figures 1 and 7.

The UAV is allowed four degrees of freedom: Translation in the x_0 and y_0 directions, denoted by x and y in the E_0 frame, rotation about the z_b axis (yaw), denoted ψ, and rotation about the x_b-axes (roll), denoted ϕ. Heading ψ is defined as the angle measured between the x_0 and x_b axes, while roll ϕ is measured between the z_0 and z_b axes, respectively. By changing both ψ and ϕ, the UAV can direct its radar signature lobes around ground-based radar sites. However, to simplify the development, the UAV is not allowed a pitching degree of freedom. Thus, the following kinematic equations will be used to model the UAV dynamic constraint (3):

$$\dot{x} = \cos\psi \tag{5}$$

$$\dot{y} = \sin\psi \tag{6}$$

$$\dot{\psi} = \frac{1}{9}\arctan u. \tag{7}$$

We constrain the roll angle ϕ to be an algebraic function of the input

$$\phi = \frac{2}{3}\arctan u. \tag{8}$$

Note that by solving (8) for $\arctan u$ and substituting into (7), one can think of the roll angle ϕ as a control input, which is constrained to satisfy $|\phi| < \pi/3$. The incorporation of

saturating function $\arctan(\cdot)$ (and the constants 2/3 and 1/9) into the dynamic equations serves to limit both the roll angle ϕ to $|\phi| < \frac{\pi}{3}$ rad $= 60°$, and the turning rate $\dot{\psi}$ to $|\dot{\psi}| < \frac{\pi}{18}$ rad$/s = 10°/s$. These are reasonable constraints for a UAV. Or, put another way, the path generated by these artificial dynamics equations should be sufficiently conservative such that any UAV can can track any solution.

As an alternative to the arctan functions, we could constrain the input using the hard bound $|u(t)| < \frac{\pi}{3}$. Although such a bound is easy to incorporate into the optimal control problem statement, and is easily handled using Pontrygin's minimum principle, it does make the subsequent numerical solution more difficult. We prefer use the softer arctan nonlinearities, and to design a cost function to have a large penalty on large u.

4.2 The Cost

The cost functional (2) should be designed to penalize a weighted sum of path length, measured radar energy at each radar site (integrated along the path), and turning rate. Thus, for both the free and fixed final time problems, we propose the following cost functional:

$$J(u) = \int_{t_0}^{t_f} \underbrace{1}_{\substack{\text{path}\\ \text{length}}} + Q \sum_{k=1}^{N} \underbrace{\frac{s(x,y,\psi,u,\bar{x}_k,\bar{y}_k)}{r(x,y,\bar{x}_k,\bar{y}_k)}}_{k^{th}\text{ radar cost}} + \underbrace{\frac{R_1}{2}u^2 + \frac{R_2}{2}\left(u - \arctan u\right)^2}_{\text{turning cost}} \, dt\,, \quad (9)$$

where $R_1 > 0$, $Q \geq 0$ and $R_2 \geq 0$ are constants, and the functions s and r are defined below. Each of these four terms requires explanation.

The first term is of course the path length, as explained above. Note that in the fixed final time problem, when t_f is known a priori, this term has no effect on the problem. That is, the optimal control that minimizes J will not be affected by the first term. Only when t_f is free, and the path length is not known a priori, does this term play a role. This should become clear when the necessary conditions are written out below.

The last two terms penalize the magnitude of the control input u. Note that the use of u in the dynamics is somewhat artificial: It is not an actual aircraft control input. However, for any real value of u, the magnitude of the roll angle $|\phi|$ is bounded by $2\pi/3$, while the magnitude of the turning rate is bounded by $\pi/18$. Thus, if u remains bounded, solutions to the dynamic equations will exist for all time. (We are ignoring the important issue of reachability here: Does there exist a control to satisfy the boundary conditions? This should be investigated as future research. It seems reasonable, however, that the answer is affirmative for sufficiently large $t_f - t_0$. This would agree with practical experience.) These terms are added to place a penalty on large roll angles, which are probably not desirable, and more importantly to make the optimal control problem well posed.

Figure 8 shows a plot of the sum of these terms for several different values of R_2 for fixed $R_1 = 1$. As can be seen in these plots, the cost is relatively "flat" until u becomes large, when the cost increases as $O(u^6)$. This is because $(u - \arctan u)^2 = (u^3/3 - u^5/5 + u^7/7 - \cdots)^2$, so the quadratic term dominates for small $|u|$, while for large $|u|$, the higher order terms in the Taylor series for $\arctan(\cdot)$ dominate.

Finally, the second term in (9) penalizes measured energy at each of the N radar sites, integrated along the path. Here, r is the distance between the UAV and the k^{th} radar site

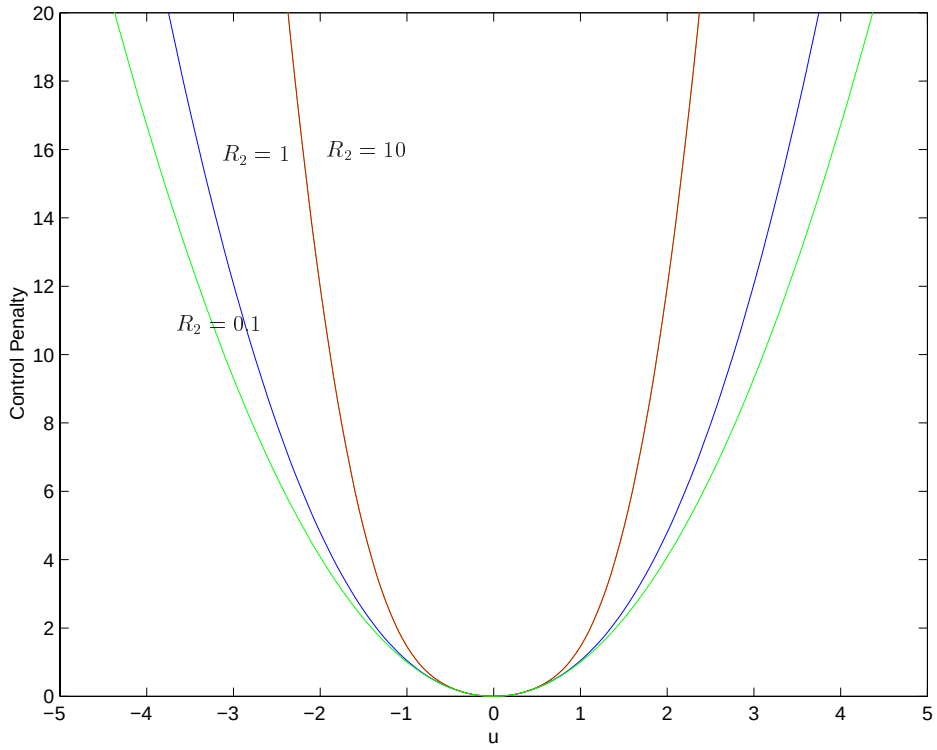

Figure 8: Control penalty (last two terms of (9)) for $R_1 = 1$ and three different values of R_2.

$(1 \leq k \leq N)$ raised to the fourth power,

$$r(x, y, \bar{x}_k, \bar{y}_k) = \left((x - \bar{x})^2 + (y - \bar{y})^2 + h^2\right)^2 , \tag{10}$$

$(\bar{x}_k, \bar{y}_k)$ are the Cartesian coordinates of the k^{th} radar site expressed in the E_0 frame, and (x, y) is the location of the UAV, also expressed in the E_0 frame. (Recall that the energy in a radar echo signal is proportional to $1/r^4$.) The numerator term s is simply the radar signature function σ discussed earlier, and captures the fact that the UAV does not radiate uniformly in all directions. However, s in (9) must be expressed in the more cumbersome Cartesian coordinates E_0 instead of the spherical E_b coordinates,

$$s(x, y, \psi, u, \bar{x}_k, \bar{y}_k) = \sigma(\hat{\mu}_k, \hat{\nu}_k) \tag{11}$$

where the conversion from spherical coordinates to Cartesian coordinates is given by

$$\hat{x}_k = (\bar{x}_k - x)\cos\psi + (\bar{y}_k - y)\sin\psi \tag{12}$$

$$\hat{y}_k = -(\bar{x}_k - x)\sin\psi\cos\phi + (\bar{y}_k - y)\cos\psi\cos\phi + h\sin\phi \tag{13}$$

$$\hat{z}_k = (\bar{x}_k - x)\sin\psi\sin\phi - (\bar{y}_k - y)\cos\psi\sin\phi + h\cos\phi, \tag{14}$$

and where

$$\hat{\mu}_k = \arctan\left(\frac{\sqrt{\hat{x}_k^2 + \hat{y}_k^2}}{\hat{z}_k}\right) \tag{15}$$

$$\hat{\nu}_k = \arctan\left(\frac{\hat{y}_k}{\hat{x}_k}\right) \tag{16}$$

and, for completeness, the definition of σ is repeated:

$$\sigma(\mu,\nu) = \begin{cases} 4\cos^2(3\nu) & \nu \le \frac{\pi}{6} \text{ rad} = 30^\circ \\ 4\cos^2(3\nu)\sin^2(2\mu) & \nu > \frac{\pi}{6} \text{ rad} = 30^\circ \end{cases} \tag{17}$$

Note that we could have allowed the radar sites to have different altitudes by changing the term h that appears in r, without any increase in complexity. Also, we could replace our "radiating" radar signature model with a reflecting radar signature model by modifying the definition of s. The "reflecting" s would depend on the same variables. However, its complexity in this case would be appreciably more complex.

4.3 Solutions

There are a number of ways to compute an optimal control $u^*(t)$ that minimizes (9) subject to (5)-(8). (Throughout this report, we use the $*$ to denote an optimal value of a particular variable.) We adopt the variational approach, which gives rise to a two-point boundary value problem, which must be solved numerically. We refer the reader to any one of a number of excellent textbooks on the subject of optimal control, e.g. [3, 2, 1].

We begin by forming the Hamiltonian function

$$\begin{aligned} \mathcal{H}(\boldsymbol{x},u,\boldsymbol{p}) &= l(\boldsymbol{x},u) + \boldsymbol{p}^T\boldsymbol{f}(\boldsymbol{x},u) && (18)\\ &= 1 + Q\sum_{k=1}^{N}\frac{s(x_1,x_2,x_3,u,\bar{x}_k,\bar{y}_k)}{r(x_1,x_2,\bar{x}_1,\bar{x}_2)} + \frac{R_1}{2}u^2 + \frac{R_2}{2}\left(u-\arctan u\right)^2 \\ &\quad + p_1\cos x_3 + p_2\sin x_3 + \frac{1}{9}p_3\arctan u, && (19)\end{aligned}$$

where $\boldsymbol{x} = [x_1\ x_2\ x_3]^T = [x\ y\ \psi]^T$ is the state, $\boldsymbol{p} = [p_1\ p_2\ p_3]^T$ is defined as the costate, and $\boldsymbol{f}$ and l are defined in (5)-(8) and (9), respectively. Computing $\dot{\boldsymbol{x}} = \frac{\partial\mathcal{H}}{\partial\boldsymbol{p}}$ and $\dot{\boldsymbol{p}} = -\frac{\partial\mathcal{H}}{\partial\boldsymbol{x}}$, we get the set of differential six equations

$$\begin{aligned} \dot{x}_1 &= \cos x_3 && (20)\\ \dot{x}_2 &= \sin x_3 && (21)\\ \dot{x}_3 &= \frac{1}{9}\arctan u^* && (22)\\ & && (23)\\ \dot{p}_1 &= -Q\sum_{k=1}^{N}\left(\frac{\partial r}{\partial x_1}(x_1,x_2,\bar{x}_k,\bar{y}_k)\cdot s(x_1,x_2,x_3,u,\bar{x}_k,\bar{y}_k)\right. && (24)\\ &\quad \left. - \frac{\partial s}{\partial x_1}(x_1,x_2,x_3,u,\bar{x}_k,\bar{y}_k)\cdot r(x_1,x_2,\bar{x}_k,\bar{y}_k)\right)/r^2(x_1,x_2,\bar{x}_k,\bar{y}_k) && (25)\\ \dot{p}_2 &= -Q\sum_{k=1}^{N}\left(\frac{\partial r}{\partial x_2}(x_1,x_2,\bar{x}_k,\bar{y}_k)\cdot s(x_1,x_2,x_3,u,\bar{x}_k,\bar{y}_k)\right. && (26)\\ &\quad \left. - \frac{\partial s}{\partial x_2}(x_1,x_2,x_3,u,\bar{x}_k,\bar{y}_k)\cdot r(x_1,x_2,\bar{x}_k,\bar{y}_k)\right)/r^2(x_1,x_2,\bar{x}_k,\bar{y}_k) && (27)\\ \dot{p}_3 &= p_1\sin x_3 - p_2\cos x_3 - Q\sum_{k=1}^{N}\frac{\partial s}{\partial x_3}(x_1,x_2,x_3,u,\bar{x}_k,\bar{y}_k)/r(x_1,x_2,\bar{x}_k,\bar{y}_k), && (28)\end{aligned}$$

which are subject to the split boundary conditions $\boldsymbol{x}(t_0) = \boldsymbol{x}_0$ and $\boldsymbol{x}(t_f) = \boldsymbol{x}_f$, and where the optimal control must satisfy

$$\mathcal{H}(\boldsymbol{x}^*, u^*, \boldsymbol{p}^*) \leq \mathcal{H}(\boldsymbol{x}^*, u, \boldsymbol{p}^*), \tag{29}$$

for any $u \in \mathcal{R}$, according to Pontrygin's minimum principle. These equations are necessary conditions for a control $u^*(t)$ and a state trajectory $\boldsymbol{x}^*(t)$ to be optimal. The idea is to solve this two point boundary value problem for a candidate optimal trajectory $\boldsymbol{x}^*(t)$, and then check to see if it is indeed a minimum of J.

There are two cases to be examined here: fixed and free final time t_f. When the final time t_f is fixed (known a priori), then the three initial conditions and three final conditions provide all the necessary boundary conditions (although they are of course split). When the final time t_f is free, i.e., unknown a priori, then we require one more equation. This comes from the fact that the Hamiltonian must remain zero along an optimal trajectory, since it is is not an explicit function of time. Thus, for the free final time case, we use the six boundary conditions in addition to the condition

$$\mathcal{H}(\boldsymbol{x}^*(t_f), u^*(t_f), \boldsymbol{p}^*(t_f)) = 0 \tag{30}$$

to compute t_f.

Equations (20)-(28) must be integrated numerically, and there are several methods to accomplish this, including so-called shooting methods, quasilinearization, and finite differences. We do not provide a general method of solution here. However, to gain some insight as to which method might work best, it is interesting to examine several special cases. We first consider the case when $Q = 0$.

4.3.1 Case 1: $Q = 0$, $R_1 > 0$, $R_2 \geq 0$.

In this case, there is no penalty on proximity to radar sites since the second term in (9) vanishes. The control appears in only the last two terms, which are positive definite. Thus, (29) can be solved for a global minimum. Computing the derivative of $\mathcal{H}$ with respect to u and setting it equal to zero gives

$$R_1 u + R_2(u - \arctan u) \cdot \left(1 - \frac{1}{1+u^2}\right) + \frac{p_3}{9}\frac{1}{1+u^2} = 0. \tag{31}$$

which must be satisfied by an optimal u^*.

Lemma 1. Equation (31) has a unique solution u^* if $R_1 >$, $R_2 \geq 0$.

Proof. Multiply (31) through by $(1 + u^2)$ and define $g_1(u) = (1 + u^2)u$ and $g_2(u) = (u - \arctan u)u^2$. Then (31) is

$$R_1 g_1(u) + R_2 g_2(u) = -\frac{p_3}{9}. \tag{32}$$

The function $g_2(u) : \mathcal{R} \to \mathcal{R}$ is 1:1 and onto, which can be seen by computing its derivative with respect to u, which is strictly positive everywhere except at the origin, where it is zero. The function $g_1(u) : \mathcal{R} \to \mathcal{R}$ is also 1:1 and onto, because its derivative is strictly positive

everywhere. Thus, the left hand side of (32) is 1:1, onto, and has a strictly positive derivative for all u. Thus, its inverse exists for any value of p_3 on the right-hand side. $\Box$

Because $Q = 0$, the Euler-Lagrange equations simplify to

$$\begin{aligned}
\dot{x}_1 &= \cos x_3 && (33)\\
\dot{x}_2 &= \sin x_3 && (34)\\
\dot{x}_3 &= \frac{1}{9}\arctan u^* && (35)\\
& && (36)\\
\dot{p}_1 &= 0 && (37)\\
\dot{p}_2 &= 0 && (38)\\
\dot{p}_3 &= p_1 \sin x_3 - p_2 \cos x_3, && (39)
\end{aligned}$$

so p_1 and p_2 are constants. In this case, we can solve (39) by integration, giving

$$p_3(t) = p_1 x_2(t) - p_2 x_1(t) \tag{40}$$

which can be verified by differentiating with respect to time and substituting (33) and (34). Thus, to compute the optimal control u^*, we need only determine the optimal constants p_1^* and p_2^*, then compute $p_3^*(t)$ using (40), and finally compute u^* using (32). These constants can be found from the boundary conditions and the fact that $\mathcal{H} = 0 \; \forall \; t \in [t_0, t_f]$, as follows. At time t_0, we know the full state $\boldsymbol{x}_0$, and can compute $u^*(p_1^*, p_2^*)$ for any value of p_1^* and p_2^*. Substituting this expression for u^* into $\mathcal{H}$, and repeating the process at t_f, gives the following two expressions

$$\begin{aligned}
\mathcal{H}(\boldsymbol{x}_0, u^*(p_1^*, p_2^*), \underbrace{[p_1^* \; p_2^* \; p_1^* x_2(t_0) - p_2^* x_1(t_0)]^T}_{\boldsymbol{p}^*(t_0)}) &= 0 && (41)\\
\mathcal{H}(\boldsymbol{x}_f, u^*(p_1^*, p_2^*), \underbrace{[p_1^* \; p_2^* \; p_1^* x_2(t_f) - p_2^* x_1(t_f)]^T}_{\boldsymbol{p}^*(t_f)}) &= 0. && (42)
\end{aligned}$$

These can be solved numerically for the optimal values p_1^* and p_2^*. Once these constants are computed, the optimal control u^* is computed by solving (32) on-line.

Although in this case the optimal trajectory will ignore the most important aspect of the path planning problem, namely the radar sites, it will prove useful as an initial condition to a more general solution. The idea is to solve a sequence of optimal control problems, defined by an increasing sequence of values of Q. The optimal solution in the case $Q = 0$ case should be close to the optimal solution when Q is small, and so whatever general numerical method is used for small $Q > 0$ should be initialized using the $Q = 0$ solution. The idea of solving a sequence of problems in this manner, parameterized by the sequence of values of Q, is called *homotopy* in the literature.

4.3.2 Case 2: $Q \geq 0$, $s = 1$, $R_1 >$, $R_2 \geq 0$.

For this case, we consider the radar signature function to be 1, so that the aircraft radiates radar energy uniformly in all directions. In this case, the Hamiltonian is

$$\mathcal{H}(\boldsymbol{x}, u, \boldsymbol{p}) = 1 + Q \sum_{k=1}^{N} \frac{1}{r(x_1, x_2, \bar{x}_1, \bar{x}_2)} + \frac{R_1}{2} u^2 + \frac{R_2}{2} (u - \arctan u)^2$$

$$+ p_1 \cos x_3 + p_2 \sin x_3 + \frac{1}{9} p_3 \arctan u, \tag{43}$$

which is a considerable simplification over the general case (18) because, in general, s depends on u. Thus, the Euler-Lagrange equations also simplify to

$$\begin{aligned}
\dot{x}_1 &= \cos x_3 \\
\dot{x}_2 &= \sin x_3 \\
\dot{x}_3 &= \frac{1}{9} \arctan u^* \\
\\
\dot{p}_1 &= \sum_{k=1}^{N} \frac{4(x_1 - \bar{x}_k)}{((x_1 - \bar{x}_k)^2 + (x_2 - \bar{y}_k)^2)^3} \\
\dot{p}_2 &= \sum_{k=1}^{N} \frac{4(x_2 - \bar{y}_k)}{((x_1 - \bar{x}_k)^2 + (x_2 - \bar{y}_k)^2)^3} \\
\dot{p}_3 &= p_1 \sin x_3 - p_2 \cos x_3,
\end{aligned}$$

while the expression for the optimal control u^* remains the same as in Case 1,

$$R_1 u^* + R_2 (u^* - \arctan u^*) \cdot \left(1 - \frac{1}{1 + u^{*2}}\right) + \frac{p_3}{9} \frac{1}{1 + u^{*2}} = 0.$$

Unfortunately in this case, the costate equations can not be solved analytically, and a numerical method will have to be brought to bear. However, the solution for the optimal control u^* is global, as in Case 1.

4.4 Case 3: The General Case.

Although we have not provided a complete solution to the two-point boundary value problem (20)-(28) in the most general case, we can propose some ideas that take advantage of the solutions provided thus far. The primary difficulty in solving the general case numerically is that the signature function s is a non-convex function of the control u. Thus, the equation $\frac{\partial \mathcal{H}}{\partial u} = 0$ used to compute the optimal control U^* as a function of state and costate will not yield a global minimum. This is not difficult to see. There is a 1:1 relationship between the control u and the roll angle ϕ. At any given fixed state (x, y, ψ), the value of s will have several minimum with respect to the variable ϕ for a fixed radar location. This can be seen by looking at Figure 2, imagining that x, y, ψ are fixed, ϕ is varied, and the reader is located at one of the fixed radar sites. This fact will considerably complicate any numerical solution.

We conjecture that the following procedure can be used to compute numerically the optimal solution in the general case.

Step 1 We first compute the optimal Voronoi solution as outlined in Section 3. This will serve as an initial condition to the procedure. Thus, the initial trajectory consists of a set of connected straight-line segments.

Step 2 The Voronoi straight-line segments will not satisfy the dynamics (5)-(7) near the end points of these segments because the magnitude of the turning rate $|\dot{\psi}|$ is bounded.

Thus, we first compute a trajectory that does satisfy the dynamics by "rounding the corners" on the Voronoi segments. This only requires fitting arcs of constant radius that correspond to the maximum turning rate at the ends of each segment.

Step 3 Next, we solve a sequence of optimal control problems for increasing values of Q, starting with a small value of Q and ending with the actual value of Q in the problem statement. Each of these problems will involve the numerical solution to a two-point boundary value problem. At each step, we use the final solution to the previous problem as the initial condition. For example, if finite elements are used, then the two-point boundary value problem reduces to the solution of a set of nonlinear equations. If Q is increased in small steps, then we can expect the solution of these equations at each step to be close to the solution at the previous step.

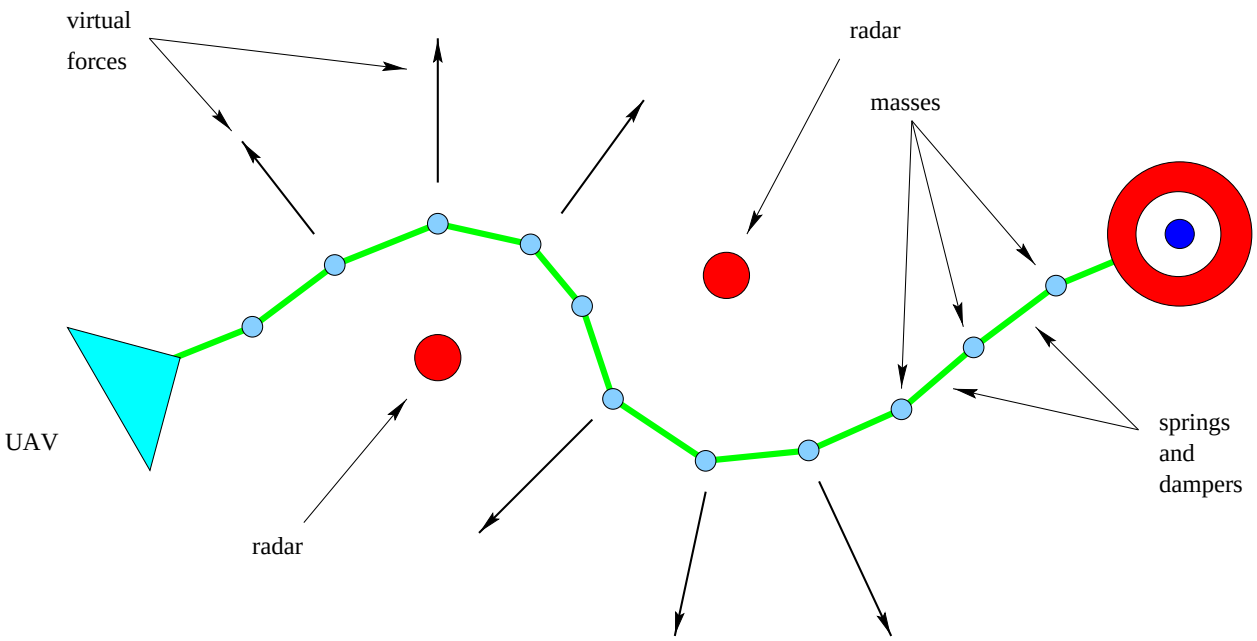

Figure 9: The chain-of-masses being acted on by both the spring restoring forces and the virtual forces pushing from each radar site.

5 Solution using Virtual Forces

The final method that we will examine in this report is a solution "by analogy," using virtual potential fields and forces. In this method, a UAV path is represented by a chain of point masses connected to one another by springs and dampers, as shown in Figure 9. One end of the chain is attached to the UAV location, while the other is attached to the target location. The chain length is parameterized by time, with t_0 denoting the present UAV location, and t_f being the final location. Just as with the optimal control solution, we assume that the UAV flies at a constant velocity, which we normalize to one without loss of generality. Again, this assumption is made only for planning purposes: The actual speed may be adjusted along the path in order to meet other requirements, e.g. rendezvous with other aircraft. The constant speed assumption simplifies the planning problem because the path length is just $t_f - t_0$. Further, we do not assume t_f is known a priori. It may be either fixed or free.

Now, if this system of springs, dampers and masses is initialized at *any* initial configuration, it will evolve in time according to the governing physics, and eventually converge to its potential energy minimum. Of course, this is a straight line of masses at zero velocity: The dampers remove all of the kinetic energy . If there were no enemy radar sites nearby, then this would be an optimal flight path between the UAV location and its target.

The key idea here is to force the chain of masses away from radar sites by using a virtual force field. To do so, assume that each radar site establishes a repulsive force field which acts to push away each mass according to an inverse-to-the-fourth law (1/distance^4). Again, if the chain is initialized in any configuration, it will eventually converge to its potential energy minimum, which is a weighted sum of its path length and the average distance from the radar sites, with zero velocity. Once it has converged, the path is defined as the sequence of way

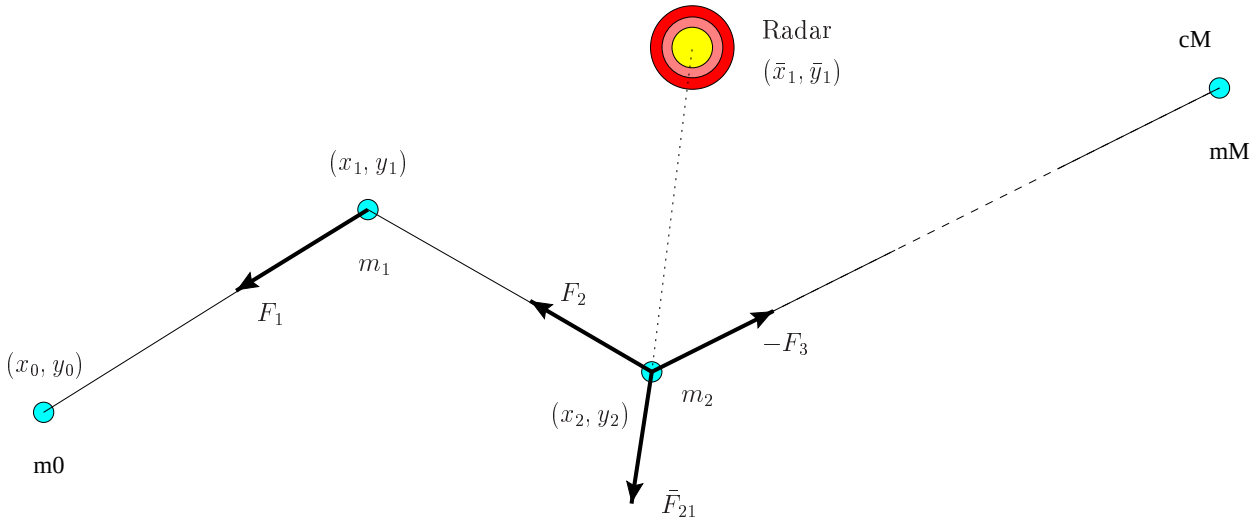

Figure 10: The chain-of-masses idea, showing the two spring forces ($\boldsymbol{F}_1$ and $-\boldsymbol{F}_2$ and one virtual "radar" force ($\bar{\boldsymbol{F}}_{21}$) acting on mass $j = 2$.

points defined by the steady-state mass locations, connected by straight line segments. The idea is illustrated in Figure 9.

We first derive the equations of motion for the uniform-radar-signature case. Referring to Figure 10, assume that we have $M+2$ masses, indexed by j $(0 \le j \le M+1)$, where the first mass $(j = 0)$ is located at the UAV location and the last mass $(j = N+1)$ is located at the target location. Let m_j denote the mass of mass j. Let (x_j, y_j) denote the (x, y)-coordinate of mass j, so (x_0, y_0) is the UAV Cartesian coordinate at time t_0, and (x_M, y_M) is the target Cartesian coordinate, i.e., the desired UAV location at time t_f. Assume the springs between the masses are linear with a spring constant of κ, and let the dampers between each of the masses have a damping constant of b.

With this notation, the distance between each of the $M + 2$ masses is

$$d_j = \sqrt{(x_j - x_{j-1})^2 + (y_j - y_{j-1})^2} \quad \text{for} \quad (1 \le j \le M+1). \tag{44}$$

Denote the normal vector pointing from mass j to mass $j-1$ as $\boldsymbol{n}_j$, so that

$$\boldsymbol{n}_j = \begin{bmatrix} (x_{j-1} - x_j)/d_j \\ (y_{j-1} - y_j)/d_j \end{bmatrix} \quad \text{for} \quad (1 \le j \le M+1).$$

Then the two spring restoring forces that act on mass j $(1 \le j \le M)$ are given by Hooke's law:

$$\begin{aligned} \boldsymbol{F}_j &= \kappa d_j \boldsymbol{n}_j \\ \boldsymbol{F}_{j+1} &= -\kappa d_{j+1} \boldsymbol{n}_{j+1}. \end{aligned}$$

The two linear viscous damping forces that act on mass j, which we denote with a $\widehat{\ }$, are similarly expressed as

$$\widehat{\boldsymbol{F}}_j = b \cdot \left(\begin{bmatrix} \dot{x}_j \\ \dot{y}_j \end{bmatrix} \cdot \boldsymbol{n}_j - \begin{bmatrix} \dot{x}_{j-1} \\ \dot{y}_{j-1} \end{bmatrix} \cdot \boldsymbol{n}_j \right) \cdot \boldsymbol{n}_j$$

$$\widehat{\boldsymbol{F}}_{j+1} \;=\; b \cdot \left(\left[\begin{array}{c} \dot{x}_{j+1} \\ \dot{y}_{j+1} \end{array} \right] \cdot \boldsymbol{n}_{j+1} - \left[\begin{array}{c} \dot{x}_{j} \\ \dot{y}_{j} \end{array} \right] \cdot \boldsymbol{n}_{j+1} \right) \cdot \boldsymbol{n}_{j+1},$$

where the $\cdot$ denotes the dot product.

Finally, the "virtual force" acting on mass j ($1 \leq j \leq M$) from radar site k ($1 \leq k \leq N$) is defined to obey an "inverse-squared-squared law." Recall that each of the N radar sites are located at $(\bar{x}_k, \bar{y}_k)$, so the distance from mass j to radar site k is

$$\bar{d}_{jk} = \sqrt{(x_j - \bar{x}_k)^2 + (y_j - \bar{y}_k)^2 + h^2}$$

where h is the height of the UAV, assumed to be constant. If we denote the normal vector pointing from radar site k to mass j as

$$\bar{\boldsymbol{n}}_{jk} = \left[\begin{array}{c} (x_j - \bar{x}_k)/\bar{d}_{jk} \\ (y_j - \bar{y}_k)/\bar{d}_{jk} \end{array} \right],$$

then we define the virtual force acting on mass j from radar k as

$$\bar{\boldsymbol{F}}_{jk} = \frac{Q}{\bar{d}_{jk}^4} \bar{\boldsymbol{n}}_{jk}$$

where Q is a constant design parameter that represents the trade-off between stealth and path length. With this, the equations of motion for each mass expressed in Cartesian coordinates is just given by Newton's law:

$$m_j \left[\begin{array}{c} \ddot{x}_j \\ \ddot{y}_j \end{array} \right] = \boldsymbol{F}_j + \boldsymbol{F}_{j+1} + \widehat{\boldsymbol{F}}_j + \widehat{\boldsymbol{F}}_{j+1} + \sum_{k=1}^{N} \bar{\boldsymbol{F}}_{jk},$$

which, expressed in Cartesian coordinates is

$$\begin{aligned} m_j \ddot{x}_j \;=\; & \underbrace{\kappa(x_{j-1} - x_j) - \kappa(x_{j+1} - x_j)}_{\text{spring forces}} + \underbrace{b(\dot{x}_{j-1} - x_j) - b(\dot{x}_{j+1} - x_j)}_{\text{damping forces}} \\ & + \sum_{k=1}^{N} \underbrace{\frac{Q}{\left((x_j - \bar{x}_k)^2 + (y_j - \bar{y}_k)^2 + h^2\right)^{5/2}}}_{\text{virtual forces}} \end{aligned} \tag{45}$$

$$\begin{aligned} m_j \ddot{y}_j \;=\; & \underbrace{\kappa(y_{j-1} - y_j) - \kappa(y_{j+1} - y_j)}_{\text{spring forces}} + \underbrace{b(\dot{y}_{j-1} - y_j) - b(\dot{y}_{j+1} - y_j)}_{\text{damping forces}} \\ & + \sum_{k=1}^{N} \underbrace{\frac{Q}{\left((y_j - \bar{y}_k)^2 + (y_j - \bar{y}_k)^2 + h^2\right)^{5/2}}}_{\text{virtual forces}} \end{aligned} \tag{46}$$

for $1 \leq j \leq M$. Note that the equations of motion for masses $j = 0$ and $j = M + 1$ are not expressed here because they are assumed to be fixed in space.

Now, to find a path, we first triangulate the region using the graph theory approaches outlined in Section 3. This produces a sequence of straight-line paths, which we use to

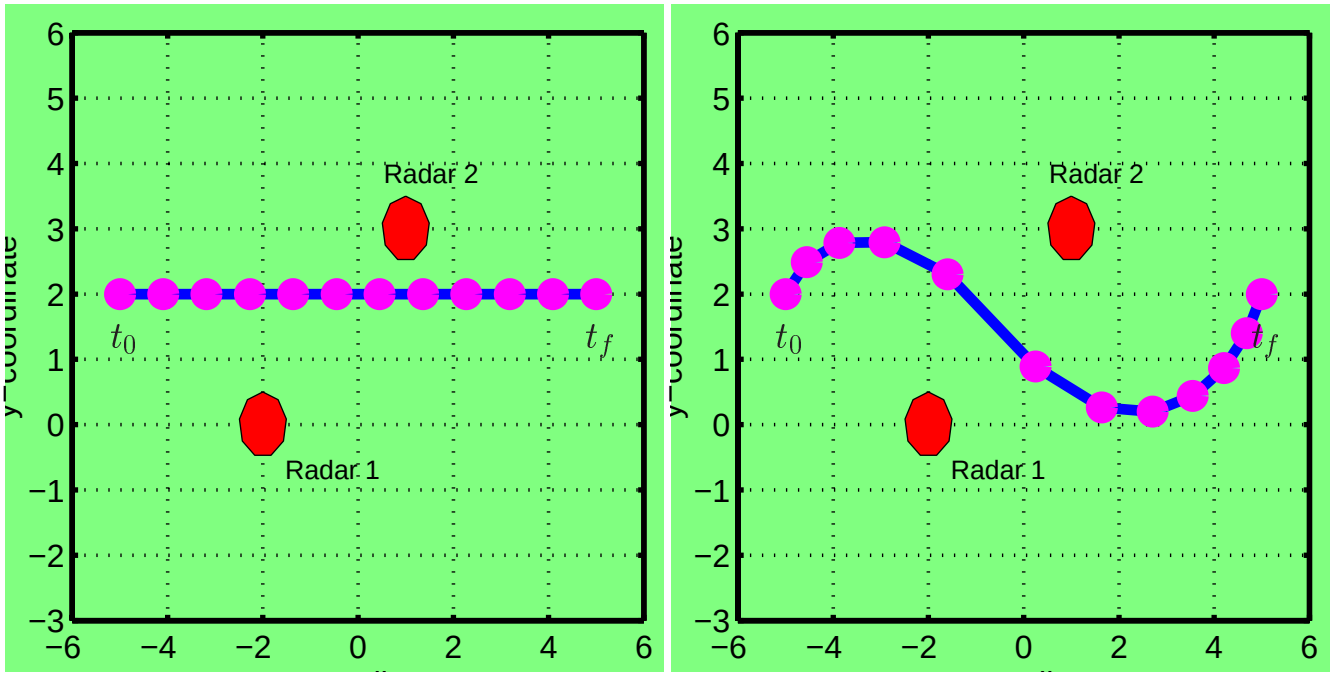

Figure 11: The chain-of-masses idea. At the top, the 10 masses ($M = 11$) are initialized along a straight line, which might be a segment from the Delaunay triangulation. The system of equations (45)-(46) is simulated and allowed to reach equilibrium, shown at bottom. For this simulation, the damping $b = 1$, the spring constant $\kappa = 1$, the masses $m_j = 1$, and the simulation time is 20s. (This should not be confused with the initial and final times t_0 and t_f.) The radar weight $Q = 50$, and the height $h = 1$.

initialize the virtual force approach. This is done by placing the M masses uniformly along these straight-lines, and then simulating the equations of motion (45)-(46) until the solution reaches equilibrium. The parameter Q is set to trade-off stealth versus path-length. If $Q = 0$, then no penalty is placed on stealth, the nonlinear terms in (45)-(46) vanish, and the masses will converge to the global straight-line equilibrium that minimizes the potential energy in the springs. For large Q, however, the masses will be pushed away from the radar sites, bending around them, reaching a local equilibrium that represents a trade-off between minimal path length and average distance from the radars. The result of a typical Matlab simulation is illustrated in Figure 11.

Several remarks are in order. First, note that the right-hand sides of (45)-(46), although nonlinear, are globally Lipschitz. This means that the solution to the differential equations will exist for all time. Moreover, note that the virtual force never larger than $1/h^4$, i.e., it "saturates" for large distances $\bar{d}_{jk}$. This means that the linear spring terms will dominate at large distances from the radars, as intuition would suggest: The path should not be affected if a radar site is very far away. From a numerical simulation point-of-view, the nonlinearity is "soft" (not severe), which makes for easy simulation of (45)-(46). This is important from the real-time implementation point-of-view.

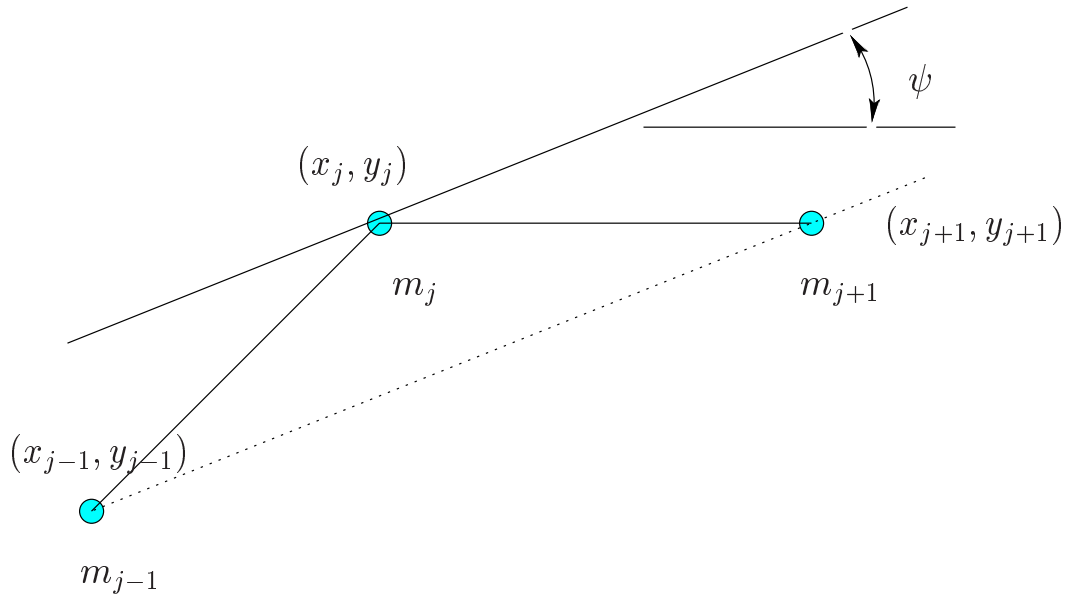

Figure 12: Assigning an orientation ψ_j to mass m_j using the approximate derivative, to incorporate the effects of a non-uniform radar signature.

5.1 Non-Convex Radar Signatures

Equations (45)-(46) must be modified to incorporate the effect of a non-uniform radar signature. For this, each mass m_j must be assigned an orientation in space, because the radar signature is a function of both yaw ψ and roll ϕ. The yaw (azimuth) orientation angle ψ_j can be defined in a discrete way for mass j by computing the angle between the x-axis and the straight line that passes through mass $j-1$ and $j+1$:

$$\psi_j = \arctan(y_{j+1} - y_{j-1},\, x_{j+1} - x_{j-1}),$$

where we have used the four-quadrant version of the inverse tangent function (`atan2` in Matlab). This is essentially the discrete approximation of the derivative of a curve, as shown in Figure 12. Once ψ_j is computed for each mass as a function of the states x_j and y_j, we can compute the roll angle ϕ_j using the kinematic (algebraic) model (5)-(8). Then the signature function $s(x_j, y_j, \psi_j, u_j, \bar{x}_k, \bar{u}_k)$, given in (11) and (8) can be evaluated, and the numerator terms in (45)-(46) can be replaced by $Q \cdot s(x_j, y_j, \psi_j, u_j, \bar{x}_k, \bar{u}_k)$.

What this modification does is make the virtual force stronger when a radar lobe is pointing toward a radar station. But, the virtual force is still bounded by $Q \max s/h^2$ because s is a globally bounded function of its arguments. Thus, solutions to (45)-(46) still exist for all time, and the modified mechanical system will again converge to a potential energy minimum. Since the virtual force is large when a lobe is pointed to a radar station, the system should converge such that the lobes point away from the stations, because the potential energy is less here. However, because the system is nonlinear, convergence will be to a local potential energy minimum; a unique global solution can not be expected to exist.

Several simulations of the non-convex case have been conducted, and Matlab simulation scripts have been included with this report for both the uniform and non-convex cases. The final "picture" for the non-convex case is similar to Figure 11.

5.2 Comparison to Optimal Control

How does the optimal control approach compare with the virtual force approach? In several aspects they are similar. This is because the virtual force method is a simulation of a Lagrangian mechanical system. In fact, the potential energy in the springs, which is given by

$$\sum_{k=1}^{N} \kappa d_i$$

is just κ times the path length, $t_f - t_0$. (Of course, this is the path length defined as the sum of the length of the straight-line segments joining the masses.) When $Q = 0$, this is the only potential energy in the system, and it is minimized, giving the straight-line solution. When $Q \geq 0$, some potential energy is "stored" in the potential field generated by the radar stations. Thus, what is minimized is a weighted sum of path length (energy stored in the springs) and distance from the radars. This is a discrete-space (lumped) version of what is being solved by the optimal control problem.

However, there are some differences between the two approaches. As shown in Section 4, the solution to the optimal control problem requires the solution to a nonlinear, non-convex two-point boundary value problem. At best, convergence to local a minimum can be expected. Numerically, the general solution can be very difficult to find. On the other hand, the virtual force approach involves computing the solution to a set of nonlinear but *stable initial value* problems. A relatively simple set of nonlinear ordinary differential equations must be simulated. Moreover, because they are globally Lipschitz and stable, the simulation is relatively straight-forward — a fixed step size can be used, for example. We can therefore expect fast convergence from *any* initial condition (the Delaunay triangulation, for example) to *a* final condition, although computing theoretical bounds on the amount of computation required will probably yield very conservative results. Note that the convergence will not be global, because the potential energy function, defined in terms of the mass positions, can be expected to possess many local minimum, especially in the non-convex radar case. But, the virtual force method is clearly superior from a computational point of view.

6 Conclusions & Recommendations

A UAV path planning algorithm should provide stealthy, minimal length paths that satisfy the dynamic constraints of the airframe. The algorithm should also be able to run on an airborne processor as a part of a larger hybrid "cooperative control" scheme. In this report, we have investigated three different designs: Graphs, Optimal Control, and Virtual Potential Fields. Of course, this report is by no means final. In fact, it is more of a proposal: There is much work to be done.

There are challenges in implementing a graph based approach. Recall that this approach discretizes the problem at the start. If stealth is the most important characteristic, then this graph must discretize the orientation degrees of freedom with sufficient fidelity at the start. For example, if there are five lobes, as in Figure 2, then a graph with only eight ($P = 8$) orientations at each location in space is probably insufficient: Only eight samples of the function $\sigma(\psi, \nu)$ as a function of ψ for a fixed value of ϕ is not enough to reconstruct

σ. In general, a radar signature with many lobes will require a large number of orientation angles at each point in space. When we add the third dimension (height) to the problem, it is not difficult to see that the number of vertices will become very large. But searching a graph with a large number of vertices becomes very memory and time consuming. This is the key challenge to this approach. Heuristic but efficient methods of searching will have to be applied, and the non-global nature of these approaches will have to be investigated. A bound on the computational complexity, in terms of time and memory requirements, will have to be developed in order to compare this with the other approaches.

If the optimal control solution is to be pursued, then the key challenge is to develop an efficient way to solve the resulting two-point boundary value problem. Using a graph-based method such as the Voronoi polygon as an initial condition to a finite-element method of solution, in combination with the homotopy ideas suggested in this report, are promising approaches. Shooting methods [8] may work well because the right-hand side of the UAV kinematic model (5)-(7) is bounded. (Shooting methods do not work well for unstable dynamical systems.) How does this compare to the graph approach? The optimal control method discretizes the problem at a later stage, when solving the continuous-state and continuous-time two-point boundary value problem. An important question to answer is how the two methods compare in terms of computational complexity. A major advantage of a graphical approach is the existence of complexity bounds, although these will be overly conservative if a heuristic search method is adopted. If the optimal control problem is reduced to the solution of a nonlinear programming problem, as it is in a finite-difference method of numerical solution, then establishing bounds on computation time should be possible. It would be interesting to compare these to a graph based method.

The potential fields approach is probably the most promising of the three. It has clear advantages: The differential equations are not only stable but are initial value problems, so mature numerical methods of solution can be applied. For a uniform radar signature, the method works very well. However, in the non-uniform case, the approach requires some more work. In this report, roll and yaw were algebraically coupled, but this does not yield satisfactory results. This can be understood by visualizing one of the masses at a point in space, with its lobe pointing toward a radar site. (See Figure 2.) This will result in a large force pushing the mass directly away from the radar site. But, the UAV could also reduce its exposure by rolling. So, if the mass were not only pushed away from the radar site, but also twisted in a direction that reduces s, the path would become more stealthy. This does not happen with the proposed model as it presently stands. However, if the potential field resulted not only in translational forces but also moments, and if roll were coupled to yaw dynamically, using a torsional spring/damper for example (and not algebraically as is done in this study), then better results should be obtained. In short, the masses need some extra degrees of freedom, and the potential field needs a "twisting" component.

Basically, the masses should be coupled by springs and dampers such that their configuration space can be interpreted as a set of way-points that the UAV can fly. Thus, the algorithm will always produce a flight path that falls within UAV flight envelope. Then the virtual potential field needs to be redefined to include a twisting component which will push the rolling degree of freedom toward a reduced potential, which is interpreted as a more stealthy way-point. To do this, the function s will have to be re-defined. Of course, these modifications will increase the number of degrees of freedom substantially, slowing down the

simulation. But very large numbers of ODEs can be simulated quickly using today's DSPs, so a large number of ODEs should not rule out this method at this point in the development cycle. Once more accurate potential field and "mass-spring-damper model" are defined, then computational complexity can be compared with the optimal control and graphical methods.

There are several other improvements and directions that can be pursued with respect to the virtual potential field approach. These are listed here.

- **Damping.** As it stands, damping is only provided along the direction of the line connecting one mass to its two neighbors. Thus, transverse modes that develop in the chain during simulation are not well-damped. This increases the time required to reach stable equilibrium. By adding damping in the transverse direction, the solution should converge faster. Notice that the additional damping does not change the potential energy in the system, so will not change the set of stable configurations. However, because we can expect the system to converge to a *local* stable equilibrium configuration, the additional damping might change *which* final equilibrium configuration is achieved.

- **Stretching.** When the chain of masses is nearby a set of radar sites, the virtual force pulls the masses away from each other, stretching out the chain. Conversely, in areas that are devoid of radar stations, the masses tend to draw close together. This affect, apparent in Figure 11, is undesirable because the masses are then interpreted as way points. It would be more desirable to have a higher density of way points near the radar sites, since one would expect the path to turn more often in these areas. So, it would seem that the approach provides its lowest resolution data in the areas where we require high resolution. We might modify the approach by using nonlinear springs, which become more stiff as the chain is pulled apart. This would minimize the stretching near radar sites. Of course, the differential equations remain stable but lose their global Lipschitz property. However, the non-Lipschitz nonlinearity is stabilizing, so should not affect the numerical solution an an overly adverse way.

- **Numerical Solution.** The ODSs have well-defined structure, and the method of numerical solution (e.g. Euler, Runge Kutta, etc.), should be chosen and tuned properly. If the improved damping is defined properly, it may be possible to define a change of coordinates in which the system is linear or almost linear. Simulation in these coordinates might be easier. It is important to establish bounds on the amount of computation required so the method can be compared with other competing approaches.

References

[1] A. E. Bryson and Y.-C. Ho, *Applied Optimal Control*. Waltham, MA: Ginn and Company, 1969.

[2] D. E. Kirk, *Optimal Control Theory: An Introduction*. Englewood Cliffs, NJ: Prentice-Hall, 1970.

[3] F. L. Lewis, *Optimal Control*. New York: John Wiley & Sons, 1986.

[4] M. Cherif, "Motion planning for all-terrain vehicles: A physical modeling approach for coping with dynamic and contact interaction constraints," *IEEE Transactions on Robotics and Automation*, vol. 15, no. 2, pp. 202–218, April 1999.

[5] T. H. Cormen, C. E. Leiserson, and R. L. Rivest, *Introduction to Algorithms*. Cambridge, MA: The MIT Press, 1990.

[6] V. Jurdjevic, *Geometric Control Theory*. New York: Cambridge, 1997.

[7] H. Sagan, *Introduction to the Calculus of Variations*. New York: McGraw-Hill, 1969.

[8] S. M. Roberts and J. S. Shipman, *Two-Point Boundary Value Problems: Shooting Methods*. New York: Elsevier, 1972.